REMARKS

The examiner is thanked for the performance of a thorough search. By this amendment, Claims 4-7, 10-13, and 16-18 are canceled, new Claims 19-31 are added, and Claims 1, 2, 3, 8, 9, and 14 are amended. Hence, Claims 1-3, 8, 9, 14, and 19-31 are currently pending in the application. Applicant respectfully requests reconsideration of the rejections, which are believed to be traversed herein.

I. ISSUES NOT RELATING TO PRIOR ART

A. **Drawings.** The Office Action Summary objects to FIG. 4 of the drawings and states that FIG. 4 should be labeled "prior art." Attached hereto is a corrected formal copy of FIG. 4 with such label; entry is respectfully requested. The Office Action also included Form PTO-948, Notice of Official Draftsperson's Drawing Review, which objected only to the margins of FIG. 7, FIG. 10. Attached hereto are corrected formal drawings for FIG. 7, FIG. 10. All the formal drawings of record are now believed to conform to the rules, and acceptance is respectfully requested.

B. **Rejection Of Claims Under 35 U.S.C. § 112.** Paragraphs 1-4 of the Office Action reject Claims 1-18 under 35 U.S.C. §112, second paragraph, as allegedly indefinite for failing to particularly point out and distinctly claim the subject matter which applicants regard as the invention. The rejection is respectfully traversed.

The Office Action contends that in claims 1-18 it is unclear what is represented by the "ED branch", "pre-integration branch" and "development branch." Specifically, the Office Action states that in Claims 1-7 the feature of "providing a branch of software source file" may be interpreted as providing a specific location (branch) in an organization for the specific software." Applicants disagree. No one of ordinary skill in the pertinent art would interpret the

feature "providing a branch of a software source code file" as referring to creating a branch in a business organization. The interpretation suggested by the Office Action is unreasonable.

The Office Action further suggests that the feature could refer to "creating a specific software code file that is an extension (branch or child file) of software that already exists (parent file). In the latter case, the branch would merely represent the extension file (child file)." Applicants disagree. The Claims are interpreted in light of the specification, and the meaning of the subject claim is apparent from the specification. For example, the term "branch" is explained in the specification in the section entitled "Additional Background Information":

> "A branch is generally, a specified configuration of source code files. The first configuration of a new release is generally designated the Main Branch or Parent Branch. Subsequently subsets or supersets of this Main Branch may be created and will be called by a specific name, such as 'Maintenance Branch' or a 'Throttle Branch' or an 'early development release' branch."

Specification, page 14, line 6. Thus, it is clear from the specification that "branch" may refer to a specified configuration of source code files. The ambiguity suggested by the Office Action does not exist.

The Office Action advances essentially the same rationale against Claims 8-18. Applicants believe that Claims 8-18 are in proper form for the foregoing reasons.

Applicants have carefully reviewed the claims to improve the readability of the claims, and as amended, Applicants believe the meanings of all claim terms are clear. Therefore, Applicants respectfully reconsideration and withdrawal of the rejection of claims 1-18 under 35 U.S.C. §112, second paragraph.

II. ISSUES RELATING TO PRIOR ART

A. **Rejection Of Claims Under 35 U.S.C. § 102.** Claims 1-3, 8-9, and 18 are rejected under 35 U.S.C. §102(b) as allegedly anticipated by *Lundin*, U.S. Patent No. 5,339,430, issued August 16, 1994. The rejections are respectfully traversed for at least the following reasons.

Claim 1, as amended, describes a release control method for providing early deployment releases of a software system. An early development code branch is established and designated for incorporation of support for new features and platforms. A plurality of pre-tested source code modules is received from a respective plurality of early integration units. Each of the pre-tested source code modules comprises one or more new features or supports one or more new platforms. The pre-tested source code modules are committed into the early development branch. Using the early development branch, a new early development release containing pre-tested source code for new features and platforms is generated.

Lundin describes a method of replacing software modules with new versions, during operation of a system, with minimal disturbance to the ongoing activities of the system. In managing the introduction of new or modified software into an operational software system, *Lundin* describes a strictly single-path linear hierarchy in which new or modified software enters the hierarchy at the lowest software level and moves serially to a higher software level in a linear migration. The new software migrates in an upward direction automatically over time, or with advanced approval by a specified review board, until it reaches the top of the hierarchy, but travels in a single path. For example, FIG. 1A of *Lundin* illustrates new software (block 4) entering the bottom of a linear hierarchy and traversing upward to final operational release at block 1.

In contrast, the present application (for example, FIG. 6) proposes an entirely separate early development branch (or technology branch) that is used to receive new source code modules before a completed, integrated branch is released. A multiple-path, treelike hierarchy, in which a plurality of pre-tested source code modules is received from a respective plurality of early integration units, as recited in Claim 1. *Lundin* has a single entry point to its hierarchy (at block 4) and no provision to receive software from multiple separate integration units or business units. Thus, *Lundin* does not teach, suggest or disclose the combination recited in Claim 1.

Indeed, *Lundin* teaches away from the present invention because its method is utilized *during* the operational process of the system. During operation of a real-time software system, it would be highly chaotic to attempt to integrate and concurrently deploy software modules from multiple hierarchical levels of integration. The entry of new and modified software into the system, formal board reviews and the upward migration of software in a non-linear configuration with multiple paths would increase the likelihood of disturbance in the system, thereby frustrating the stated purpose of *Lundin*.

The Office Action further contends that "*Lundin* also indicates that his System provides a branch (creates a Parent file from an existing file), see Col. 6 lines 47-56 and col. 7 lines 2-11 (hierarchical...levels – branch)." Applicants disagree. Col. 6, lines 47-56 describes generic class inheritance principles that are a common feature of object-oriented programming languages. This is not the same as providing an early development code branch that is designated for incorporation of support for new features and platforms, and receiving a plurality of pre-tested source code modules from a respective plurality of early integration units, as claimed in Claim 1.

Further, the "hierarchies" described by *Lundin* in fact are linear lists and not tree-like hierarchies having multiple sources, as proposed by Applicants. In its argument, the Office Action improperly assumes an ambiguity in Applicants' claim terminology that does not exist.

As described above, an example meaning of the term "branch" is clear from the specification, and *Lundin* does not have an equivalent disclosure.

The Office Action further contends that *Lundin* shows the claim feature of "using the ED branch, generating a new early development release containing pre-tested source code for new features and platforms," citing col. 10, lines 29-53. This is incorrect. The cited portion of *Lundin* describes using a series of pointers to direct traffic from a test version of software to an old version of software, and use of dynamic runtime binding. Nothing in the cited portion describes, teaches or suggests using an early development branch of code to generate a new release with pre-tested source code, as claimed.

Claims 2 and 3 each depend on Claim 1, and therefore include all the steps and limitations of Claim 1. Thus, Claims 2 and 3 are believed to be allowable for the same reasons set forth above with respect to Claim 1. Accordingly, Applicants respectfully submit that Claims 2 and 3 are allowable for the reasons given above with respect to Claim 1.

Claims 8 and 9 are expressed in the format of a system that executes the steps of Claims 2 and 3, which depend on Claim 1, and therefore include all the steps and limitations of Claim 1. Thus, the reasons set forth above with respect to Claim 1 apply fully to Claims 8 and 9. Accordingly, Applicant respectfully submits that Claims 8 and 9 are allowable for the reasons given above with respect to claim 1.

With respect to Claim 18, Applicants disagree with the rationale of the Office Action; however, to expedite prosecution, Claim 18 is canceled, and therefore the contentions of the Office Action are not further addressed herein.

For the foregoing reasons, Applicants respectfully request reconsideration and withdrawal of the rejection under §102 with respect to Claims 1-17.

B. **Rejection Of Claims Under 35 U.S.C. § 103.** In paragraph 5 of the Office Action, Claims 4-7, 10-13, and 14-18 are rejected under 35 U.S.C. §103 as allegedly unpatentable over *Lundin* as applied to Claim 2, and further in view of *Hutton*. The rejections are respectfully traversed for at least the following reasons.

As a threshold matter, all of Claims 4-7, 10-13, and 14-18 depend, directly or indirectly, from one of the independent claims that are discussed above with respect to *Lundin*. Because each of the independent claims has at least one feature not found in *Lundin*, a combination of *Lundin* with *Hutton* cannot reach the claimed subject matter. Therefore, the rejection under 103 should be withdrawn for the reasons given above with respect to *Lundin*.

Claim 4 recites the release control method of claim 2 wherein the regular recurring basis is approximately every 8 weeks. With respect to Claim 4, the Office Action contends that *Hutton* suggests the features of Claim 4 by describing that "small program fixes and enhancements are released about every two weeks." This is incorrect. The preamble of Claim 1, from which Claim 4 depends indirectly, recites a release control method for providing early deployment releases of a software system, in which the early deployment releases contain support for new features and platforms. Those of ordinary skill in the art know that releases that contain support for new features and platforms are not "small program fixes and enhancements." *Hutton* is concerned with bug fixes and the like of far smaller scope than the new support contemplated by Claim 4.

With respect to Claims 5-7, 11-13, and 16-18, Applicants disagree with the rationale of the Office Action; however, to expedite prosecution, Claims 5-7, 11-13, and 16-18 are canceled and therefore the statements of the Office Action are not addressed further herein.

Regarding Claim 14, the Office Action states that the selection step is inherent. As amended, Claim 14 further features testing the selected features in a plurality of business units, providing the selected features to a pre-integration branch only when testing in the business units

is successful, testing the features in the pre-integration branch, and providing the features to a development branch only when testing in the pre-integration branch is successful. Neither *Lundin* nor *Hutton* discloses, suggests, or teaches such steps. Accordingly, Claim 14 is believed to be in allowable condition.

Regarding Claim 15, the Office Action states, "Claim 15 is rejected as claim 4 in view of claim 3." This is improper. The Office may not reject a claim in view of another claim. A rejection requires the Office to establish a *prima facie* case of unpatentability for a claim by relying on prior art references or other proper evidence. The Office has failed to do so with respect to Claim 15. Further, because there is no technical basis or prior art basis asserted with respect to Claim 15, Applicants have no way to respond to the rejection. Reconsideration and withdrawal of the rejection of Claim 15 are respectfully requested.

III. CONCLUSIONS

The above amendment is submitted to more particularly describe or claim the Applicant's invention and is not made to distinguish over any known reference or prior art. Therefore, based in at least the reasons stated above, it is respectfully submitted that Claims 1-18 are allowable over the art of record and in condition for allowance.

To the extent necessary, a petition for extension of time under 37 C.F.R. §1.136 is hereby made. A check is enclosed for all fees that are believed to be necessary in connection with this response. However, the Commissioner is hereby authorized to charge any shortage in fees due in connection with the filing of this paper, including extension of time fees, to Deposit Account 50-1302 and to credit any excess fees to such deposit account.

Should any additional issues remain that might be resolved by interview or Examiner's amendment, the Examiner is invited to contact the undersigned at telephone number (408) 414-1213.

Respectfully submitted,

HICKMAN PALERMO TRUONG & BECKER LLP

Dated: August 6, 2002

By: _____
Christopher J. Palermo
Registration. No. 42,056

1600 Willow Street
San Jose, California 95125-5106
Telephone No.: (408) 414-1080
Facsimile No.: (408) 414-1076

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Box Amend-No Fee, Commissioner for Patents, Washington, D.C. 20231

on August 6, 2002 by _____
Teresa Austin

VERSION WITH MARKINGS TO SHOW CHANGES MADE

1. (Amended) A release control method for providing early deployment releases of a software system, the early deployment releases containing support for new features and platforms, the method comprising the steps of:

a. providing [a] an early development branch of the software [source code file] system that is designated for incorporation of one or more software modules providing support for new features and platforms [hereinafter called an ED branch];

b. receiving, from a plurality of integration units, a plurality of pre-tested software modules, wherein each of the pre-tested software modules comprises one or more new features or supports one or more new platforms;

[b]c. committing the pre-tested [source code] software modules for new features and platforms into the [ED] early development branch; and

[c]d. using the [ED] early development branch, generating a new early development release containing pre-tested [source code] software modules for new features and platforms.

2. (Amended) The release control method of claim 1 comprising the additional step of repeating steps [b and c] c and d on a regular recurring basis for a fixed number of cycles.

3. (Amended) The release control method of claim 1 wherein the pre-tested software module is received at a pre-integration branch that is separate from the early development branch, and wherein the committing step comprises committing pre-tested [source code] software modules for new features and platforms from a pre-integration branch into the [ED] early development branch.

[4. (Canceled) The release control method of claim 2 wherein the regular recurring basis is approximately every 8 weeks.]

[5. (Canceled) The release control method of claim 2 wherein the fixed number of cycles is 5 or more.]

[6. (Canceled) The release control method of claim 2 wherein repeated releases created by repeating steps b and c on a regular recurring basis for a fixed number of cycles comprise steps of generating a fat release followed by a thin release alternating in such a fashion throughout the fixed number of cycles.]

[7. (Canceled) The release control method of claim 6 wherein alternating releases are releases of new version of an internetworking operating system (IOS).]

8. (Amended) A system for providing early deployment releases of a software system, the early deployment releases containing support for new features and platforms, comprising:

a. [a] an early development branch of the software [source code file] system designated for incorporation of one or more software modules providing support for new features and platforms [hereinafter called an ED branch];

b. logic for receiving, from a plurality of integration units, a plurality of pre-tested software modules, wherein each of the pre-tested software modules comprises one or more new features or supports one or more new platforms;

[b]c. logic for committing the pre-tested [source code] software modules for new features and platforms into the [ED] early development branch;

[c]d. using the [ED] early development branch, logic for generating a new early development release containing pre-tested [source code] software modules

for new features [and platform] or platforms on a regular recurring basis for a fixed number of cycles; and

[d]e. logic for generating said new early development release containing pre-tested [source code] software modules for new features [and platform] or platforms on a regular recurring basis for a fixed number of cycles.

9. (Amended) The system of claim 8 wherein the logic for committing comprises logic for committing pre-tested [source code] software modules for new features and platforms from a pre-integration branch into the [ED] early development branch.

[10. (Canceled) The system of claim 8 wherein the regular recurring basis is approximately every 8 weeks.]

[11. (Canceled) The system of claim 8 wherein the fixed number of cycles is 5 or more.]

[12. (Canceled) The system of claim 8 wherein the logic for generating said new early development release containing pre-tested source code for new features and platform on a regular recurring basis for a fixed number of cycles comprises logic for generating a fat release followed by a thin release alternating in such a fashion throughout the fixed number of cycles.]

[13. (Canceled) The system of claim 12 wherein alternating releases are releases of new versions of an internetworking operating system (IOS).]

14. (Amended) A product release method for controlling the release of software system code based on a fixed frequency, the method comprising the steps of:

a. selecting one or more features for inclusion in a new release of the software system code base, wherein a quantity of features selected will allow a

next scheduled release of the software system code base to be completed at a required time;

b. [committing] testing the quantity of features selected [into a pre-integration branch of the software system code base; and] in a plurality of business units;

c. [testing] providing the quantity of features selected [in the pre-integration branch so as to commit the quantity of features selected into a development branch on time to allow the next scheduled release of the software system code to be completed at the required time] to a pre-integration branch of the software system code base only when testing in the business units is successful;

d. testing the quantity of features selected in the pre-integration branch;

e. providing the quantity of features selected to a development branch only when testing in the business units is successful and in time to allow the next scheduled release of the software system code base to be completed in the required time.

15. (Unamended) The method of claim 14 comprising the additional steps of:

a. completing testing of a modified software system code base in the development branch which contains the quantity of features selected and tested in the pre-integration branch; and

b. releasing the modified software system code base at the required time.

[16. (Canceled) The method of claim 15 wherein steps a through e are repeated in a regular recurring basis for a fixed number of cycles.]

[17. (Canceled) The method of claim 16 wherein repeated releases created by repeating steps a through e on a regular recurring basis for a fixed number of cycles comprise steps of generating a fat release followed by a thin release alternating in such a fashion throughout the fixed number of cycles.]

[18. (Canceled) The method of claim 16 wherein the regular recurring basis is approximately every 8 weeks.]

19. (New) A method as recited in Claim 1, further comprising the steps of:
receiving and testing a plurality of software source code modules that support new features or platforms at a respective plurality of business unit pre-integration branches;
committing one or more of the plurality of software source code modules from the one or more of the business unit pre-integration branches to a central pre-integration branch only when such testing is successful; and
committing the plurality of software source code modules from the central pre-integration branch to the early development branch when all the modules have been committed from the business unit pre-integration branches to the central pre-integration branches.

20. (New) A method as recited in Claim 19, further comprising the step of generating, using the early development branch, a new early development release containing pre-tested source code for new features and platforms only when the plurality of software source code modules has been committed from the central pre-integration branch to the early development branch.

21. (New) A method as recited in Claim 1, further comprising the steps of:
receiving a plurality of software source code modules that support new features or platforms at a respective plurality of business unit pre-integration branches;
at each business unit, testing each feature of the software source code modules of that business unit individually, in combination with each other feature individually, and in combination with all other features;

committing one or more of the plurality of software source code modules from the one or more of the business unit pre-integration branches to a central pre-integration branch only when such testing is successful; and

committing the plurality of software source code modules from the central pre-integration branch to the early development branch when all the modules have been committed from the business unit pre-integration branches to the central pre-integration branches.

22. (New) A method as recited in Claim 19, further comprising the step of generating, using the early development branch, a new early development release containing pre-tested source code for new features and platforms only when the plurality of software source code modules has been committed from the central pre-integration branch to the early development branch.

23. (New) A computer-readable medium comprising one or more stored sequences of instructions for providing release control using early deployment releases of a software system, the early deployment releases containing support for new features and platforms, which instructions, when executed by one or more processors, cause the one or more processors to perform the steps of:

a. providing an early development branch of a software release that is designated for incorporation of support for new features and platforms;

b. receiving, from a plurality of integration units, a plurality of pre-tested source code modules, wherein each of the pre-tested source code modules comprises one or more new features or supports one or more new platforms;

c. committing the pre-tested source code for new features and platforms into the early development branch; and

d. using the early development branch, generating a new early development release containing pre-tested source code for new features and platforms.

24. (New) A computer-readable medium as recited in Claim 23, further comprising the steps of:

receiving and testing a plurality of software source code modules that support new features or platforms at a respective plurality of business unit pre-integration branches;

committing one or more of the plurality of software source code modules from the one or more of the business unit pre-integration branches to a central pre-integration branch only when such testing is successful; and

committing the plurality of software source code modules from the central pre-integration branch to the early development branch when all the modules have been committed from the business unit pre-integration branches to the central pre-integration branches.

25. (New) A computer-readable medium as recited in Claim 24, further comprising the step of generating, using the early development branch, a new early development release containing pre-tested source code for new features and platforms only when the plurality of software source code modules has been committed from the central pre-integration branch to the early development branch.

26. (New) A computer-readable medium as recited in Claim 23, further comprising the steps of:

receiving a plurality of software source code modules that support new features or platforms at a respective plurality of business unit pre-integration branches;

at each business unit, testing each feature of the software source code modules of that business unit individually, in combination with each other feature individually, and in combination with all other features;

committing one or more of the plurality of software source code modules from the one or more of the business unit pre-integration branches to a central pre-integration branch only when such testing is successful; and

committing the plurality of software source code modules from the central pre-integration branch to the early development branch when all the modules have been committed from the business unit pre-integration branches to the central pre-integration branches.

27. (New) A computer-readable medium as recited in Claim 24, further comprising the step of generating, using the early development branch, a new early development release containing pre-tested source code for new features and platforms only when the plurality of software source code modules has been committed from the central pre-integration branch to the early development branch.

28. (New) A system as recited in Claim 8, further comprising the steps of:

receiving and testing a plurality of software source code modules that support new features or platforms at a respective plurality of business unit pre-integration branches;

committing one or more of the plurality of software source code modules from the one or more of the business unit pre-integration branches to a central pre-integration branch only when such testing is successful; and

committing the plurality of software source code modules from the central pre-integration branch to the early development branch when all the modules have been committed from the business unit pre-integration branches to the central pre-integration branches.

29. (New) A system as recited in Claim 28, further comprising the step of generating, using the early development branch, a new early development release containing pre-tested source code for new features and platforms only when the plurality of

software source code modules has been committed from the central pre-integration branch to the early development branch.

30. (New) A system as recited in Claim 8, further comprising the steps of:

receiving a plurality of software source code modules that support new features or platforms at a respective plurality of business unit pre-integration branches;

at each business unit, testing each feature of the software source code modules of that business unit individually, in combination with each other feature individually, and in combination with all other features;

committing one or more of the plurality of software source code modules from the one or more of the business unit pre-integration branches to a central pre-integration branch only when such testing is successful; and

committing the plurality of software source code modules from the central pre-integration branch to the early development branch when all the modules have been committed from the business unit pre-integration branches to the central pre-integration branches.

31. (New) A system as recited in Claim 8, further comprising the step of generating, using the early development branch, a new early development release containing pre-tested source code for new features and platforms only when the plurality of software source code modules has been committed from the central pre-integration branch to the early development branch.